TU WIEN Informatics

Visibility Precomputation with RTX Ray Tracing

Thomas Bernhard Koch
Visual Computing

TU Wien Informatics
Institute of Visual Computing & Human-Centered Technology
Research Unit of Computer Graphics
Supervisor: Univ.Prof. Dipl.-Ing. Dipl.-Ing. Dr.techn. Wimmer Michael
Contact: thomas-koch@kabelplus.at

Introduction

Visibility is a fundamental problem in computer graphics and computer vision and is typically computed from a point or from a region.From-region visibility is often precomputed due to its higher computational complexity. Some approaches utilize well-known real-time rendering techniques, such as the z-buffer, while other methods use ray casting to determine the visibility of objects and primitives, such as the from-region visibility approach *Guided Visibility Sampling (GVS)* [1].
In this work, we are revisiting Guided Visibility Sampling to find out how efficient it can become when hardware-accelerated ray tracing is used. Furthermore, we improve upon it and present a new approach, called *Guided Visibility Sampling++*. Our approach uses hardware-accelerated ray tracing and is over four orders of magnitude faster than the original CPU-based implementation of GVS.

Contributions

Our contributions can be summarized as follows:

> Guided Visibility Sampling++ (GVS++), an aggressive from-region visibility sampling approach. Our approach improves upon Guided Visibility Sampling. GVS++ is more accurate and offers better parallelizability than GVS. This makes an efficient implementation, where the main sampling workload is executed on the GPU, possible.

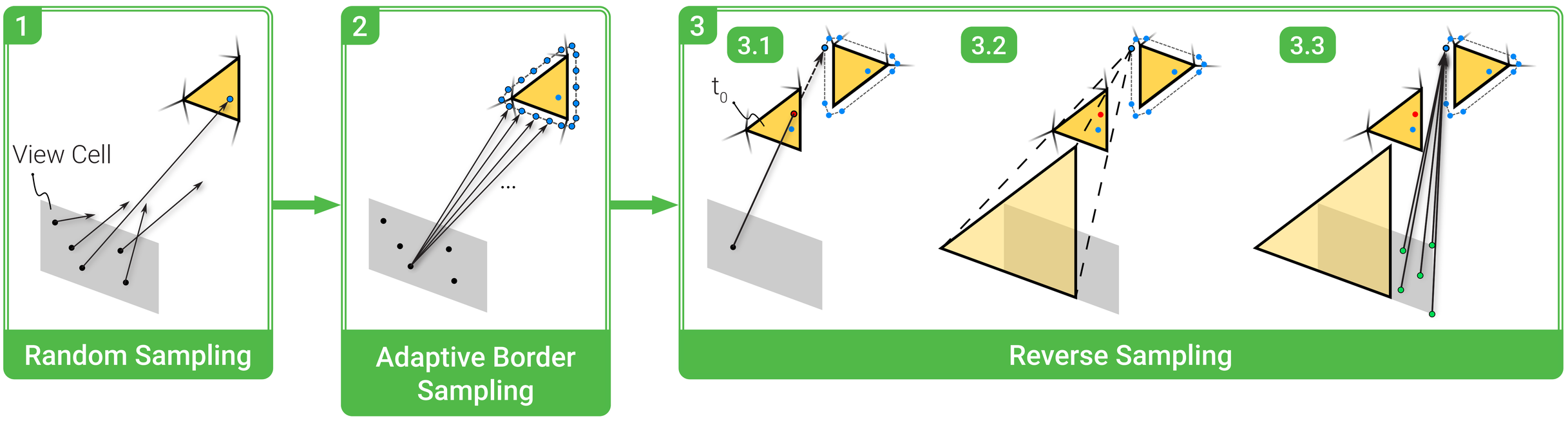
> A publicly available Vulkan implementation of GVS++ that uses hardware-accelerated ray tracing.

> An in-depth analysis of GVS++ on multiple scenes and a comparison to a brute-force random sampling approach, a rasterization-based from-region visibility technique, and GVS.

Method - Guided Visibility Sampling++

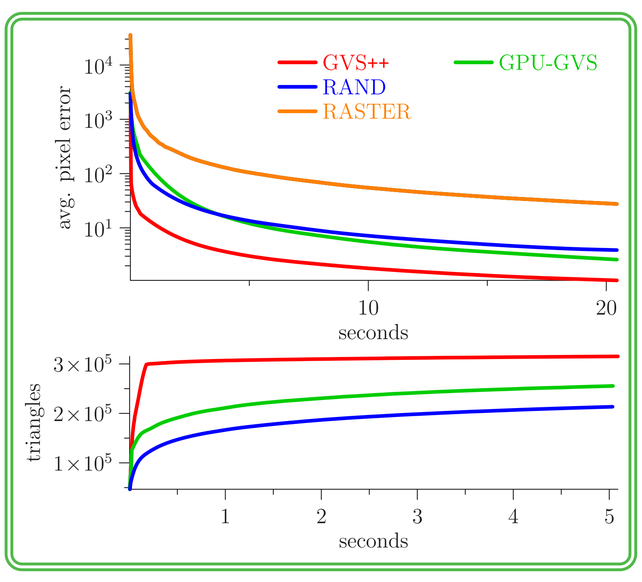
Our approach combines multiple intelligent sampling strategies to efficiently construct a set of visible triangles, called potentially visible set (PVS):

1. An initial set of triangles is constructed by randomly sampling the scene.
2. *Adaptive Border Sampling* samples along the borders of previously found triangles to find adjacent triangles.
3. *Reverse Sampling* samples discontinuities that can occur while a triangle is processed by the Adaptive Border Sampling algorithm. A discontinuity is detected if an intersected triangle t_0 is closer than the current target sample location (3.1). Reverse Sampling then computes new ray origins on the view cell from which the discontinuity can be sampled without intersecting t_0. To get such positions, Reverse Sampling projects t_0 onto the view cell (3.2) followed by uniformly distributing points on the view cell that are not within the projected triangle (3.3). These points are then used as mutated ray origins to sample the discontinuity.

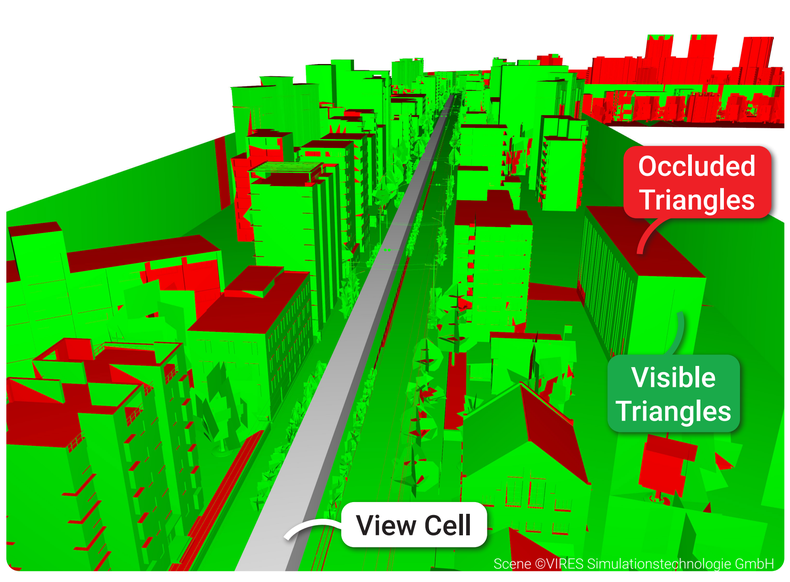


The right data structures and optimizations are crucial for the performance of our algorithm. We store the PVS on the GPU and only transfer it to the host once our algorithm terminates. This allows our ray-casting shaders to distinguish whether a triangle is intersected for the first time. This way, no expensive set operations on the host-side are necessary. A naïve implementation where set operations on a host-side PVS are executed is over 20x slower than our optimized implementation.

Results



Our approach calculates a more accurate solution in a shorter timespan than brute-force random sampling (RAND), GVS, and a comparable rasterization-based visibility sampling technique (RASTER). GVS++ typically computes a well-converged PVS under one second. This result is attributed to our efficient sampling strategies that enable efficient parallel execution on the GPU.
A significant performance difference can be observed when comparing our Vulkan GVS++ implementation with hardware-accelerated ray tracing to the original CPU-based implementation of GVS, where GVS++ is over four orders of magnitude faster than CPU-based GVS.



[1] Peter Wonka, Michael Wimmer, Kaichi Zhou, Stefan Maierhofer, Gerd Hesina, and Alexander Reshetov. 2006. Guided visibility sampling. ACM Transactions on Graphics (TOG) 25, 3 (2006), 494–502